

10

```
FOR (i=0; i<2; i=i+1)
  begin
    out[i] = 8'b10101010;
    enable[i] = up[2*i];
  end
```

FIGURE 1

```
reg y [3:0];  
WHILE (x <= y)  
  begin  
    fpl_bit[x+y] = mm_iru[x-y];  
  end
```




FIGURE 2

FOR (INIT; EXIT; INC)
begin
BODY_OF_STATEMENTS;
end

The diagram shows a FOR loop structure with five handwritten annotations and arrows: 16 points to 'INIT', 18 points to 'EXIT', 20 points to 'INC', 14 points to the opening parenthesis of the loop header, and 22 points to the 'begin' keyword.

FIGURE 3

```
out[0] = 8'b10101010;  
enable[0] = ~up[0];  
  
out[0] = 8'b10101010;  
enable[0] = ~up[2];
```

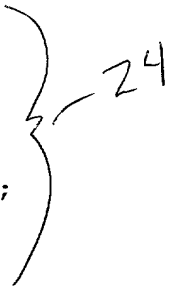


FIGURE 4

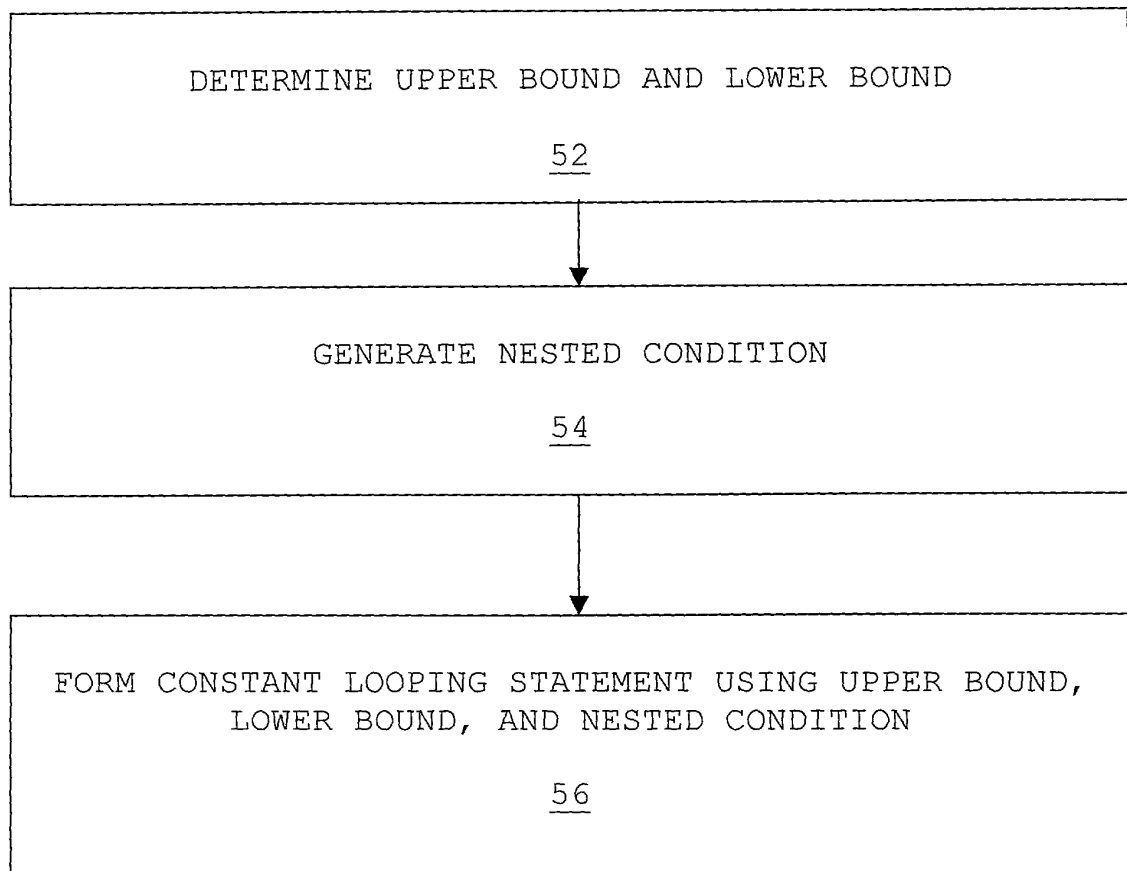


FIGURE 5

T03050"0T0500

60 { FOR (LOWER_BOUND_EXPRESSION; UPPER_BOUND_EXPRESSION;
INCREMENT_EXPRESSION) 66
if (INIT && EXIT) 68
STATEMENT_BODY 69

FIGURE 6

80
82 reg i [3:0];
84 reg j [1:0];
reg k [2:0];
for (j<=i; i<k; i=i+1)
statement_body

74 76
70 72 78

FIGURE 7

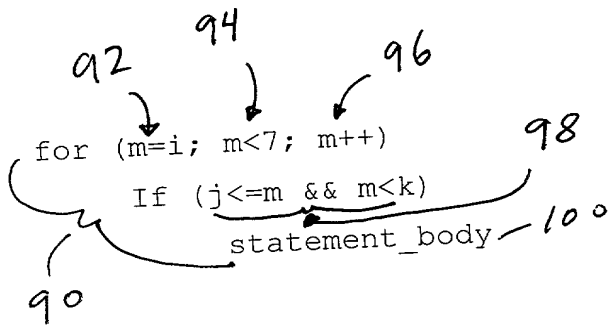


FIGURE 8


```
WHILE ( x <= 15 )  
  if ( x <= y )  
    begin  
      fpl_bit[x+y] = mm_iru[x-y];  
    end  
end
```

The flowchart illustrates the execution of a WHILE loop. Handwritten annotations include: '112' pointing to the loop condition '(x <= 15)'; '114' pointing to the 'if (x <= y)' condition; '116' pointing to the assignment statement 'fpl_bit[x+y] = mm_iru[x-y];'; and '110' pointing to the 'end' statement, which indicates the loop's termination point.

FIGURE 9